NAVEGAÇÃO AUTÔNOMA USANDO TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL E IMAGENS

Ana Paula Abrantes de Castro
Aluna da Universidade Braz Cubas, Bolsista PIBIC/CNPq
Orientador: MSc. José Demisio Simões da Silva, LAC

As redes neurais são sistemas paralelos distribuídos compostos por unidades de processamento que computam determinadas funções matemáticas (lineares ou não). O funcionamento das redes neurais tem como inspiração a estrutura física do cérebro humano. Nas soluções de problemas por redes neurais, estes são representados nos padrões de conexão da rede, e o paralelismo inerente aos sistemas de redes neurais, criam a possibilidade de um desempenho superior ao dos modelos convencionais. Uma das áreas de maior aplicação das redes neurais é o reconhecimento de padrões. Entretanto, as redes neurais também têm sido usadas em controle de sistemas de forma eficiente. A navegação autônoma tem sido objeto de vários estudos e pesquisas na área de Inteligência Artificial, como pode ser constatado pelos resultados nas copas mundiais de futebol jogado por robôs (Robocups) e nas diversas publicações em revistas e jornais especializados. Ela constitui uma área de grandes possibilidades de uso de sistemas de redes neurais e/ou de lógica nebulosa. O trabalho em desenvolvimento consiste na implementação de um modelo computacional de navegação adaptativa auxiliada por redes neurais ou lógica nebulosa, para um objeto ou um modelo de objeto móvel, que será dotado com capacidade adaptativa durante a navegação. Para o caso de um modelo de objeto móvel, o sistema deverá aprender a navegar em um ambiente definido, utilizando uma rede neural, uma composição de redes neurais diferentes, lógica nebulosa ou uma hibridização de redes neurais com lógica nebulosa. Qualquer uma das técnicas deverá ser capaz de manter o objeto na trajetória especificada. No caso de um objeto móvel este deverá apresentar autonomia, devendo navegar no ambiente de forma adaptativa, com realimentação por imagens. Com isso, o objeto terá capacidade de se autoguiar em um ambiente, corrigindo sua trajetória de forma automática a partir da informação extraída das imagens, através de técnicas de visão computacional. O sistema de IA guiará o objeto para que ele permaneça na trajetória real em que se desloca. Como resultado do trabalho, objetiva-se conseguir um modelo de navegação utilizando técnicas de IA.

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS - INPE

RELATÓRIO PARCIAL DE BOLSA DE INICIAÇÃO CIENTÍFICA

Ana Paula Abrantes de Castro
Universidade Braz Cubas, UBC,
Bolsista PIBIC/CNPq Processo No. 101884/99/0
Período: 04/1999 a 06 /1999
E-mail: apaula@lac.inpe.br

Título: **ESTUDO DE NAVEGAÇÃO AUTÔNOMA UTILIZANDO CONTROLE ADAPTATIVO IMPLEMENTADO COM TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL.**

Orientador: MsC. José Demisio Simões da Silva - LAC/INPE

Bolsa de Iniciação Científica suportada pelo Conselho Nacional de Pesquisa e Desenvolvimento - CNPq, desenvolvida no Laboratório de Computação e Matemática Aplicada – LAC/INPE.

Título: **ESTUDO DE NAVEGAÇÃO AUTÔNOMA UTILIZANDO CONTROLE ADAPTATIVO IMPLEMENTADO COM TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL.**

Ana Paula Abrantes de Castro
Universidade Braz Cubas, UBC,
Bolsista PIBIC/CNPq Processo No. 101884/99/0
Período: 04/1999 a 06 /1999
E-mail: apaula@lac.inpe.br

Orientador: MsC. José Demisio Simões da Silva - LAC/INPE
São José dos Campos, 29 de Junho de 1999.

RESUMO

As redes neurais são sistemas paralelos distribuídos compostos por unidades de processamento que computam determinadas funções matemáticas (lineares ou não). O funcionamento das redes neurais tem como inspiração a estrutura física do cérebro humano. Nas soluções de problemas por redes neurais, estes são representados nos padrões de conexão da rede, e o paralelismo inerente aos sistemas de redes neurais, criam a possibilidade de um desempenho superior ao dos modelos convencionais. Uma das áreas de maior aplicação das redes neurais é o reconhecimento de padrões. Entretanto, as redes neurais também têm sido usadas em controle de sistemas de forma eficiente. A navegação autônoma tem sido objeto de vários estudos e pesquisas na área de Inteligência Artificial, como pode ser constatado pelos resultados nas copas mundiais de futebol jogado por robôs (Robo cup) e nas diversas publicações em revistas e jornais especializados. Ela constitui uma área de grandes possibilidades de uso de sistemas de redes neurais e/ou de lógica nebulosa. O trabalho em desenvolvimento consiste na implementação de um modelo computacional de navegação adaptativa auxiliada por redes neurais ou lógica nebulosa, para um objeto ou um modelo de objeto móvel, que será dotado com capacidade adaptativa durante a navegação. Para o caso de um modelo de objeto móvel, o sistema deve aprender a navegar em um ambiente definido, utilizando uma rede neural, uma composição de redes neurais diferentes, lógica nebulosa ou uma hibridização de redes neurais com lógica nebulosa. Qualquer uma das técnicas deve ser capaz de manter o objeto na trajetória especificada. No caso de um objeto móvel este deve apresentar autonomia, devendo navegar no ambiente de forma adaptativa, com realimentação por imagens. Com isso, o objeto tem capacidade de se autoguiar em um ambiente, corrigindo sua trajetória de forma automática a partir da informação extraída das imagens, através de técnicas de visão computacional. O sistema de IA guia o objeto para que ele permaneça na trajetória real em que se desloca. Como resultado objetiva-se conseguir um modelo de navegação utilizando técnicas de IA.

SUMÁRIO

	Pág.
1. INTRODUÇÃO	4
2. REDES NEURAIS	6
2.1 Neurônios Biológicos	7
2.2 Neurônios Artificiais	8
2.3 Funcionamento das Redes Neurais Artificiais	9
2.4 Características das Redes Neurais	11
2.5 Tipos de Redes Neurais	13
2.6 Treinamento de uma rede neural de camadas múltiplas por backpropagation	17
3. LÓGICA NEBULOSA	19
4. EXEMPLOS DE SISTEMAS	22
5. IMPLEMENTAÇÃO	24
6. CONCLUSÕES E PERSPECTIVAS	25
BIBLIOGRAFIA	
ANEXO	

1. INTRODUÇÃO

Uma das tarefas mais difíceis nas pesquisas em Inteligência Artificial (IA) tem sido dotar robôs com habilidade para operar no mundo real (Jochem e Pomerleau, 1996), em especial em tarefas que envolvem navegação autônoma. Assim, a navegação autônoma tem sido objeto de vários estudos e pesquisas na área de IA, como pode ser visto pelos por trabalhos como Jochem e Pormeleau (1996), Buhmamm et al. (1995), Shem et al. (1997), Sargent et al. (1997), Kitano et al. (1997), Tyengar e Thomas, (1992), Yang et al. (1995), Osuma e Luo, (1995) e Thrun, (1997), entre outros, constituindo-se uma área de grandes possibilidades de uso de sistemas de redes neurais e/ou de lógica nebulosa.

Em geral, o objetivo das pesquisas é dotar o sistema de robô (ou qualquer outro veículo que possa navegar) com capacidade adaptativa ao ambiente por onde navega, mantendo sua trajetória inicial. Neste sentido existem duas possibilidades de controle adaptativo: uma quando a trajetória a ser percorrida é pré definida e a outra quando a trajetória é totalmente desconhecida.

Assim, muitas pesquisas têm se valido de métodos e técnicas de IA, como Sistemas Especialistas, Redes Neurais e Lógica Nebulosa, no desenvolvimento de subsistemas de controle que permitam a navegação em uma trajetória conhecida e que tomem decisão de replanejamento da trajetória em função de informações que recebem do ambiente através de sensores, agindo sobre atuadores do sistema do veículo como na figura 1, apresentando portanto um comportamento adaptativo.

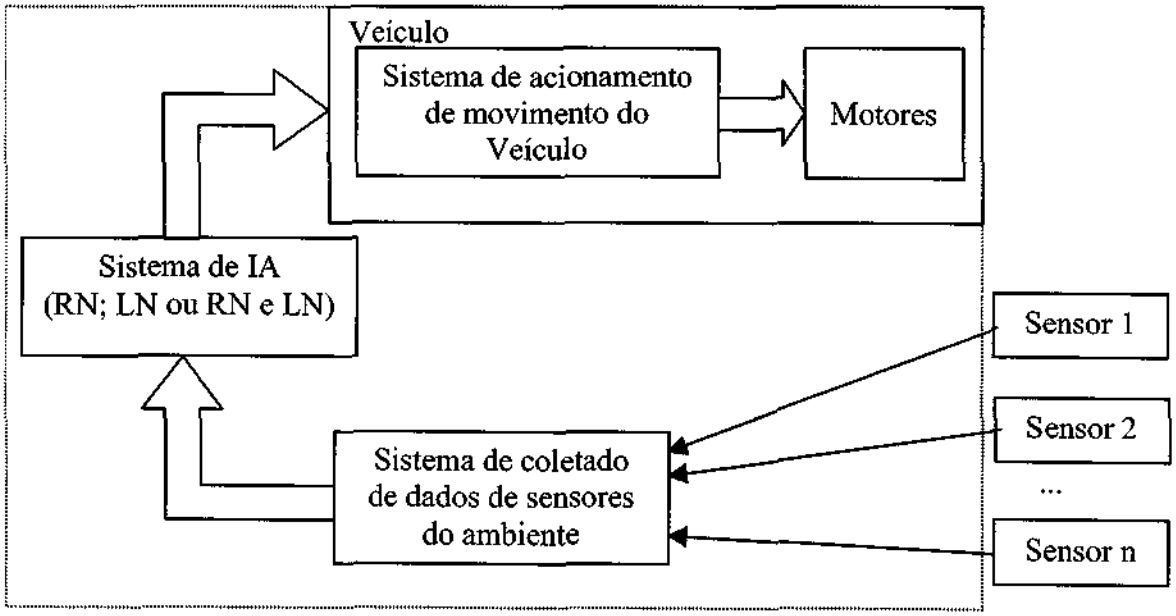

Figura 1 – Esquema genérico de navegação autônoma utilizando técnicas de IA.

Em se tratando de sistemas que têm uma realimentação do ambiente para replanejar sua trajetória em tempo de execução, faz-se necessário utilizar sensores que permitam a inferência de informações do ambiente em que o veículo navega. Para isto diversos tipos de sensores podem ser utilizados como por exemplo, sonar, sensores infravermelho, sensores de temperatura, câmeras, etc.. Em geral todos os sensores devem fornecer dados suficientes para que o sistema de controle calcule a distância de possíveis obstáculos para a posição do veículo. Com isso o sistema de controle pode tomar decisão de redirecionamento da trajetória evitando o choque com os obstáculos. Outra tarefa de navegação consiste em manter o veículo em uma trajetória inicial que a princípio pode ser desconhecida (Jochem e Pormeleau, 1996)

O uso de câmeras, além de se mostrar como meio natural de entrada de dados para o sistema de IA, prover uma grande quantidade de informação do ambiente. Entretanto, para usar imagens em tarefas de tempo real faz-se necessário usar algoritmos rápidos de visão computacional para extração de características a serem utilizadas na inferência sobre a tarefa a ser realizada para manter o veículo na trajetória. Alguns trabalhos já mostram resultados satisfatório de como obter informações e como realimentá-la no sistema, como por exemplo Jochem e Pormeleau, (1996).

O trabalho em desenvolvimento consiste na implementação de um modelo computacional de navegação adaptativa auxiliada por redes neurais ou lógica nebulosa, para um objeto móvel real ou um modelo de objeto móvel, que será dotado com capacidade adaptativa durante a navegação. Em se tratando de um modelo de objeto móvel, o sistema deverá aprender a navegar em um ambiente definido utilizando uma rede neural, uma composição de redes neurais diferentes, lógica nebulosa ou uma hibridização de redes neurais com lógica nebulosa. Qualquer uma das técnicas deverá ser capaz de manter o objeto na trajetória especificada. Para o caso de um objeto móvel, este deverá apresentar autonomia, devendo navegar no ambiente de forma adaptativa, com realimentação por imagens. Com isso, o objeto terá capacidade de se movimentar em um ambiente, corrigindo sua trajetória de forma automática a partir da informação extraída das imagens utilizando técnicas de visão computacional. Como resultado do trabalho, objetiva-se conseguir um modelo de navegação utilizando técnicas de IA, que possam ser utilizadas em outras situações de forma rápida, sem envolver grandes modificações na estrutura do veículo.

Um dos problemas associados com o uso de imagens está na escolha dos operadores de visão computacional que extrairão as características das imagens para realimentar no sistema.

Este relatório apresenta resultados parciais do projeto iniciado em meados de abril de 1999. Portanto, as seções seguintes são divididas em: seções 2 e 3 tópicos sobre redes neurais e lógica nebulosa, respectivamente, escolhidas como as técnicas de IA que deverão ser usadas no desenvolvimento do sistema; A seção 4 apresenta alguns desenvolvimentos de trabalhos com navegação autônoma; a seção 5 comenta sobre a implementação do projeto em desenvolvimento, objeto deste relatório; e a seção 6 apresenta as perspectivas de trabalho para a continuação do projeto.

2. REDES NEURAIS

O cérebro humano é um sistema extremamente complexo, capaz de pensar, aprender, reconhecer objetos, tomar decisões e realizar outras atividades que são executadas espontaneamente. Estas são as principais motivações para o estudo de inteligência artificial, ou seja, estudar métodos inspirados no funcionamento do cérebro, para que os computadores realizem tarefas semelhantemente aos humanos.

A área de redes neurais também é conhecida por neurocomputação, redes associativas, sistemas paralelos e conexionistas. A tecnologia de redes neurais é inspirada nos sistemas nervosos biológicos, e se utiliza dos modelos matemáticos da cognição humana ou da biologia neural. Semelhantemente aos sistemas biológicos, as redes neurais têm a capacidade de aprender através de dados fornecidos.

McCulloch e Pitts, em 1943, apresentaram um modelo da atividade dos neurônios biológicos. Usando elementos lógicos simples, mostraram como redes feitas de muitas dessas unidades interconectadas poderiam realizar operações lógicas. Rosemblatt, em 1958 desenvolveu uma rede neural denominada "perceptron". Widrow e Hoff, em 1960, conceberam a "ADALINE" com um combinador linear adaptativo. Depois de um período de pouca pesquisa na área na década de 70, após a publicação do livro *"Perceptrons"* de Minsky e Papert em 1969, as pesquisas em redes neurais ressurgiram com a publicação do artigo de Hopfield em 1982. Desde então as atividades na área têm crescido exponencialmente, justificando a criação de jornais especializados (Pao, 1989; Lisboa, 1992).

2.1. Neurônios Biológicos

O cérebro é composto de aproximadamente 10^{11} neurônios de diversos tipos diferentes que são conectados a outros 10^4 neurônios.

A célula nervosa, ou neurônio, foi identificada no século XIX pelo neurologista Ramón Cajal. O neurônio possui um corpo celular, ou soma, que é o centro dos processos metabólicos da célula nervosa, a partir dele projetam-se extensões filamentares, os dendritos e o axônio (Figura 2). Os dendritos cobrem um volume muitas vezes maior que o próprio corpo celular e formam uma árvore dendrital. O axônio conecta a célula nervosa à outras. Em geral o neurônio possui um único axônio, embora esse possa apresentar ramificações (Kovács, 1996).

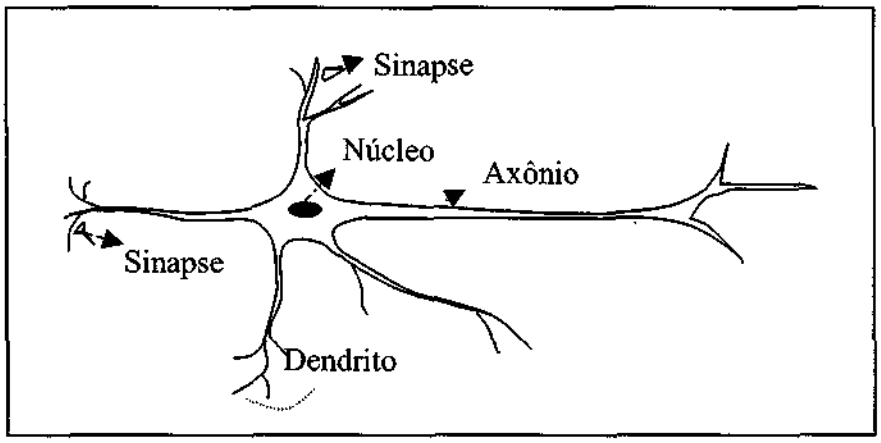

Figura 2 - Esquema da Célula neural

O neurônio biológico pode ser visto como sendo o dispositivo computacional elementar básico do sistema nervoso, possuindo muitas entradas e uma única saída. As entradas ocorrem através das conexões sinápticas, que conectam a árvore dendrital aos axônios de outras células nervosas. Os sinais que chegam por estes axônios são pulsos elétricos conhecidos como impulsos nervosos, e constituem a informação que o neurônio processará, para produzir como saída um impulso nervoso no seu axônio.

As sinapses são regiões eletroquímicas ativas entre duas membranas: a pré-sináptica por onde chega um estímulo proveniente de uma outra célula; e a pós-sináptica no dendrito. Na região intersináptica, o estímulo nervoso que chega à sinapse é transferido para a membrana dendrital através dos neurotransmissores, que dependendo do tipo, originarão uma conexão excitatória ou inibitória. Uma conexão excitatória provoca uma alteração no potencial da membrana que contribui para a formação de um impulso nervoso no axônio de saída, enquanto que uma conexão inibitória age no sentido oposto (Kovács, 1996).

2.2. Neurônios Artificiais

Os componentes de um neurônio artificial têm uma analogia direta com os componentes dos neurônios biológicos. A Figura 3 mostra a representação esquemática de um neurônio artificial.

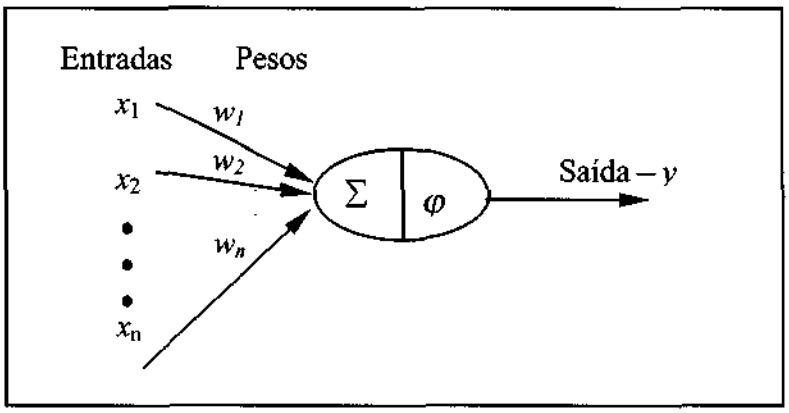

Figura 3 – Representação de um neurônio artificial.

Assim como os neurônios biológicos possuem muitas entradas, dadas pelos níveis de estímulos, os neurônios artificiais também têm inúmeras entradas que são apresentadas simultaneamente. Cada uma dessas entradas é modificada por um peso (ou peso sináptico), que são representados por w_1, w_2, ..., w_n. Os pesos são coeficientes adaptativos dentro da rede, e determinam a intensidade do sinal da entrada. Cada entrada é multiplicada por um peso relativo, que afeta o seu valor, semelhantemente à junção sináptica dos neurônios biológicos. Os pesos podem ser positivos (excitatórios) ou negativos (inibitórios). Os sinais de entrada depois de ponderados pelos pesos, são somados. Matematicamente esse processo pode ser dado pela equação:

$$y_j = \varphi\left(\sum_{k=1}^{n} w_{jk} x_k\right) \tag{1}$$

As entradas e os pesos são os vetores $\boldsymbol{x} = (x_1, x_2 \ldots x_n)$ e $\boldsymbol{w} = (w_1, w_2 \ldots w_n)$, respectivamente. Portanto, o sinal de entrada total da função φ é o produto interno dos dois vetores, ou seja, a soma da multiplicação de cada componente do vetor $\boldsymbol{x}$ pelo seu correspondente do vetor $\boldsymbol{w}$. O resultado é um escalar, que geometricamente indica uma medida de similaridade entre os dois vetores. Se os vetores apontam para a mesma direção, o produto interno é máximo, e se os vetores apontam em direção oposta, o produto interno é mínimo.

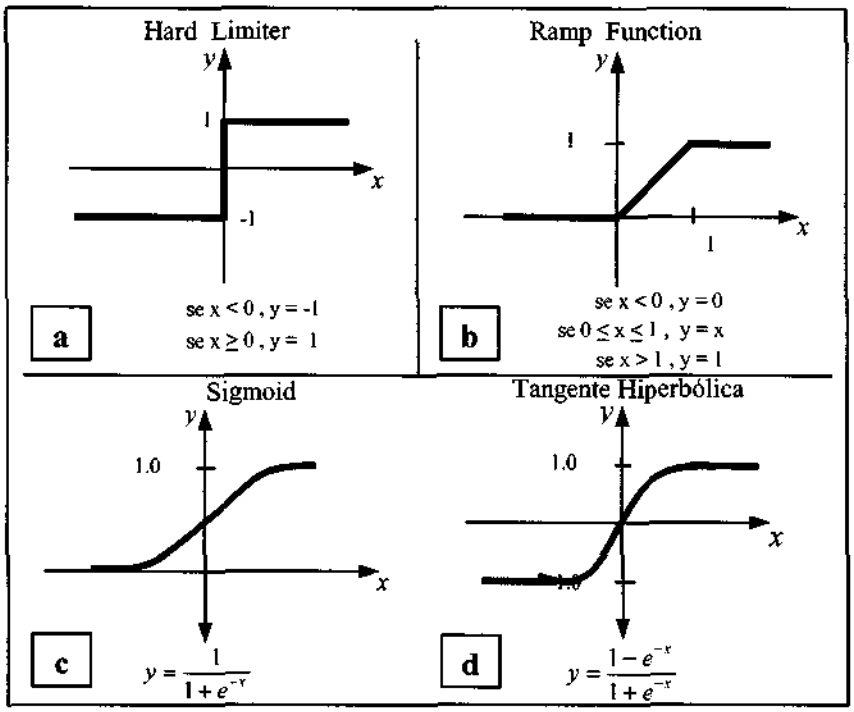

Figura 4 - Tipos de Funções de transferência.

A função φ é responsável pelo disparo do neurônio. A Figura 4 apresenta alguns tipos de funções de ativação φ. As funções lineares têm saída proporcional às entradas. A função "hard limiter" ou degrau, tem saída igual a 1 ou –1, ou 1 ou 0, ou seja, sempre saídas binárias (Figura 4.a). A função de transferência do tipo rampa (Figura 4.b) reflete a entrada dentro de uma determinada faixa linear, mas funciona como um limite rígido ("hard limiter") fora dela. As funções sigmoid e tangente hiperbólica, (Figuras 4.c e 4.d), são funções contínuas não lineares, que são preferidas em alguns modelos de redes para resolver problemas não lineares.

2.3. Funcionamento das redes neurais artificiais

As redes neurais resultam da combinação dos neurônios em camadas, que podem ter conexões totais ou parciais. A rede é dita totalmente conectada se todas as saídas de uma camada são passadas para todos os neurônios na próxima camada (Figura 5).

A camada que recebe a entrada é chamada de camada de entrada; as outras são chamadas camadas "escondidas" porque são internas à rede e não têm um contato direto com o meio externo e a última camada, que é a camada de saída.

Quando não há conexões laterais entre os neurônios artificiais, numa dada camada e não há realimentação para a camada anterior, a rede é chamada de "feed-forward". Em todos os casos, essas conexões têm pesos que têm que ser treinados.

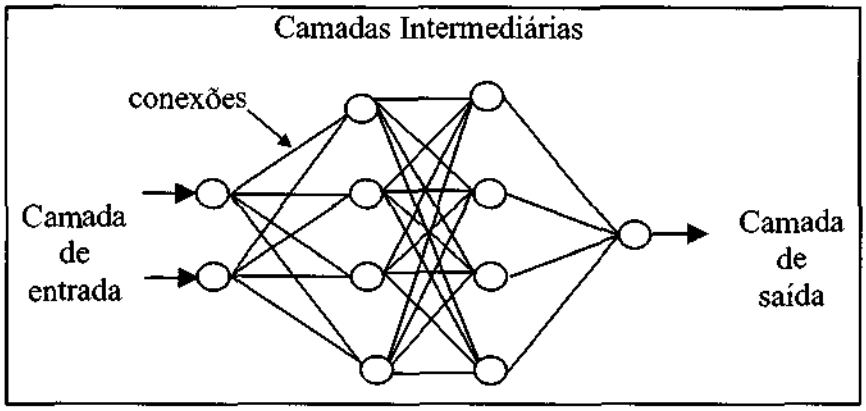

Figura 5 - Rede simples com 4 camadas.

Cada conexão entre os neurônios artificiais têm um peso ajustável, como mostrado na Figura 6. Este é um exemplo de uma rede neural totalmente conectada, do tipo feed-forward com 3 entradas, 4 neurônios na camada oculta e 2 neurônios na camada de saída.

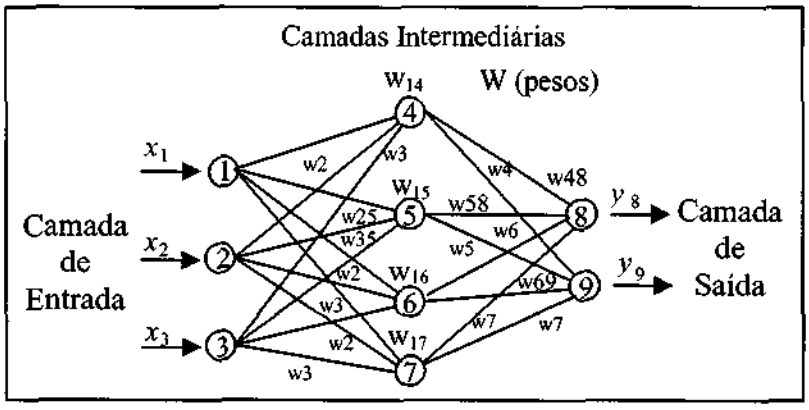

Figura 6 - Rede Neural com 3 Camadas

A camada de entrada recebe o vetor de entrada ***X***, cujos componentes são x_1, x_2, x_3, e após a ativação da rede se obtém o vetor de saída ***Y***, com componentes y_8 e y_9. Cada componente do vetor de entrada é distribuído para todos os neurônios da camada subsequente. Observando por exemplo o componente x_1 na figura 6, este é aplicado na entrada e vai para cada um dos neurônios artificiais na camada oculta, passando através dos pesos respectivos pesos w_{14}, w_{15}, w_{16} e w_{17}. O mesmo ocorre para os componentes do sinal de entrada x_2 e x_3, que também vão para os neurônios 4, 5, 6 e 7, através dos respectivos pesos.

Observa-se que o neurônio 4 da camada interna, tem três entradas provenientes da camada de entrada que têm seus valores modificados pelas conexões dos pesos w_{14}, w_{24} e w_{34}. Este neurônio processa a informação que recebe como descrito pela equação (1). Assim, a primeira etapa é somar os 3 sinais recebidos. O resultado da soma é passada para a segunda etapa que é a aplicação de uma função de transferência, como mostrado na Figura 4, a saída desta função de transferência é apresentada aos neurônios 8 e 9 da camada de saída através dos pesos w_{48} e w_{49}. Os neurônios 5, 6 e 7 têm o mesmo processamento que o neurônio 4. Os neurônios 8 e 9 coletam as entradas multiplicadas pelos pesos, e somam os valores, passando através da função de transferência para produzir y_8 e y_9, os componentes do vetor de saída Y.

Resumindo as redes neurais artificiais como na Figura 6, operam de uma maneira simples e direta. Quando o vetor X é aplicado à entrada da camada, a multiplicação das entradas pelos pesos, bem como a soma e a passagem pela função de transferência são rapidamente executados de camada a camada até a saída. Entretanto quando há conexões de realimentação, o processo é muito mais complicado, pois a rede neural recebe informação de volta a partir da saída.

2.4. Características das Redes Neurais

O desenvolvimento de aplicações utilizando redes neurais divide-se em duas fases: a aprendizagem e a ativação. A aprendizagem é o processo de adaptar os pesos das conexões em resposta às entradas. A ativação consiste no processo de receber uma entrada e produzir uma saída de acordo com a estrutura da rede que foi treinada. A ativação da rede também é parte integrante do processo de aprendizagem que requer a ativação da rede, para poder comparar a saída obtida com a saída desejada.

As regras de aprendizagem diferem na maneira como os pesos das conexões devem ser ajustados de acordo com a resposta da ativação, da entrada ou do erro entre a saída da ativação e a resposta correta fornecida.

As regras de aprendizagem podem ser classificadas em supervisionadas e não-supervisionadas. Na aprendizagem supervisionada a rede neural é treinada para apresentar uma resposta desejada à uma entrada específica e os pesos da rede são alterados em função da diferença para a resposta correta. No aprendizado não supervisionado não há uma saída específica, a resposta é baseada na habilidade da rede se auto organizar internamente para que cada neurônio escondido responda a diferentes conjuntos de entrada.

A maioria das aplicações de redes neurais na engenharia envolvem o aprendizado supervisionado. Com a apresentação dos vetores de entrada e de saída desejada, é possível calcular a diferença entre a saída desejada e a resposta atual constituindo o erro, que pode ser usado para ajustar a conexão dos pesos, em se tratando de aprendizado por correção do erro. Em outros casos, os pesos são ajustados de acordo com critérios preestabelecidos pela natureza do processo de aprendizado, como um aprendizado competitivo ou "hebbiano" . Em qualquer caso, o aprendizado consiste em calcular a variação do peso (Δw) que deve ser aplicada aos pesos atuais da rede, de forma que os novos pesos podem ser calculados pela regra delta:

$$w_{ij}^{novo} = w_{ij}^{velho} + \Delta w_{ij} \tag{2}$$

Os métodos de aprendizagem diferente na maneira como calcular o delta, ou seja a segunda parcela do lado direito da equação (2).

Para demonstrar o aprendizado supervisionado, a rede neural apresentada na Figura 7 é modificada incluindo uma saída desejada, um comparador e um algoritmo de ajuste de peso onde a saída desejada é representada pelo vetor $\boldsymbol{Z}$ com componentes z_8 e z_9. As entradas para o comparador são: o padrão de saída desejado $\boldsymbol{Z}$ é a saída calculada pela rede, $\boldsymbol{Y}$. O erro que vem do comparador, que é a diferença entre $\boldsymbol{Y}$ e $\boldsymbol{Z}$, é utilizado no algoritmo de ajuste de pesos para determinar o valor do ajuste que deve ser feito nos pesos da rede para tentar reduzir o erro. O processo de apresentação dos vetores de entrada e de saída desejada, com a ativação da rede, o cálculo do erro e alteração dos pesos em função do erro, é repetido até que o erro seja reduzido ao objetivo ou um número máximo de iterações seja atingido. Quando o vetor de saída $\boldsymbol{Y}$ e o vetor desejado $\boldsymbol{Z}$ são equivalentes, pode-se dizer que a rede foi treinada para mapear o vetor de entrada $\boldsymbol{X}$ num vetor de saída $\boldsymbol{Z}$, essa é a essência do treinamento supervisionado.

O uso da aprendizagem caracteriza uma característica importante das redes: aprendizagem por exemplos apresentados. Tipicamente essa forma constitui um modelo que representa o relacionamento entre as variáveis de entrada e saída. As redes neurais são distributivas e associativas, a distribuição significa que a informação é propagada entre os pesos que foram ajustados no processo de treinamento. Essas conexões (pesos) são as unidades de memória das redes, e os valores dos pesos representam o estado atual do

conhecimento da rede, desde que, cada unidade individual de conhecimento seja distribuída sobre todas as unidades de memória da rede.

As redes neurais também são tolerantes à falhas, elas podem aprender a tomar decisões com dados incompletos (Tsoukalas e Uhrig, 1997). Como o conhecimento é distribuído ao longo do sistema, uma porcentagem das entradas pode faltar sem ocorrer mudanças significativas no comportamento do sistema.

A computação das redes neurais consiste em interconectar as unidades que agem instantaneamente nos dados de forma paralela. Os computadores digitais atuais em geral apenas simulam esse paralelismo. O paralelismo pode ser explorado com redes neurais implementadas em hardware, permitindo decisões extremamente rápidas em tempo real.

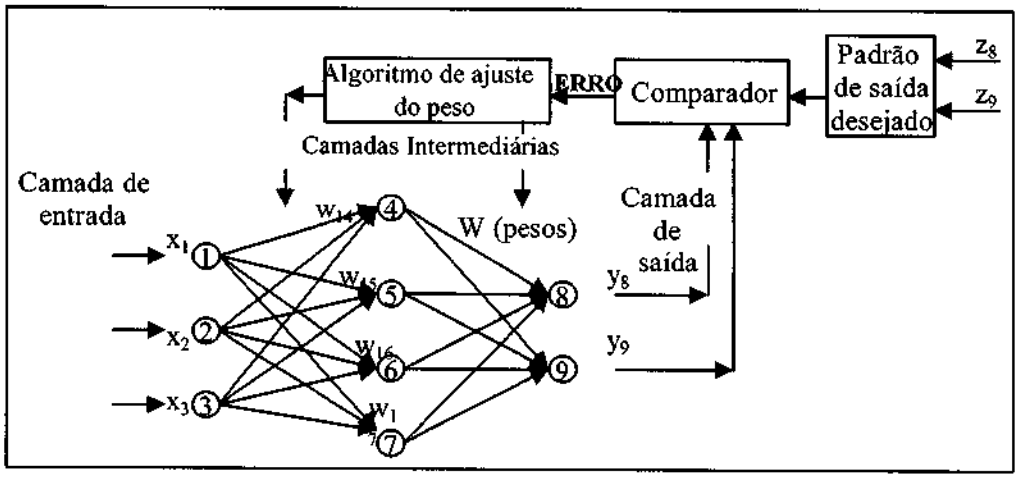

Figura 7 - Treinamento de uma rede feedforward.

2.5. Tipos de Redes Neurais

A seguir serão apresentados alguns modelos de redes neurais mais comuns na literatura.

Perceptron

O perceptron (Figura 8), é o mais antigo paradigma de rede neural implementado como um modelo computacional da retina. A rede consiste de 2 camadas conectadas, onde a de entrada é uma camada de sensores que são projetados para a camada de saída. A saídas podem ser -1 ou 1, ou seja a função de ativação é do tipo hard limiter (Figura 4). Este tipo de função de saída exige que os padrões sejam linearmente separáveis para serem classificados.

Assim, problemas não lineares como OU-exclusivo por exemplo, não podem ser resolvidos, demonstrando uma limitação do percetpron.

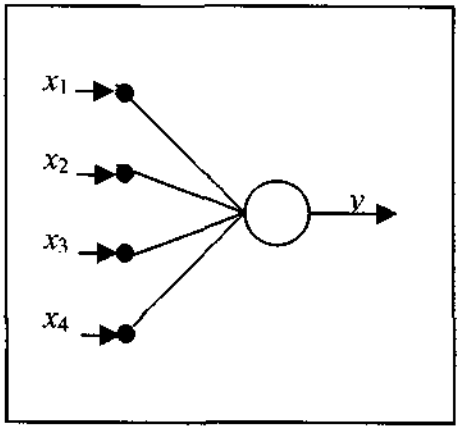

Figura 8 – Rede neural do tipo Perceptron.

No entanto, o Perceptron pode manipular certas classes de problemas muito bem, incluindo classificação de formas, reconhecimento de caracteres, e uso em sistemas de visão computacional. O Perceptron tem sido o bloco de construção fundamental para modelos mais poderosos, como a rede Adaline.

O algoritmo do perceptron é apresentado a seguir:

1) Inicialização do vetor $w(0) = 0$ (pode ser arbitrário)

2) Ativação.
No tempo t ($t = 1, 2, ...$) aplicar o vetor de entrada $x(t)$ com valores contínuos e a resposta desejada.

3) Computar a resposta atual (da rede)
$y(t) = \text{sgn}(v(t)) = \text{sgn}(w^T(t) * x(t))$
$$\text{onde sgn}(v) = \begin{cases} +1 \text{ se } v \geq 0 \\ -1 \text{ caso contrário} \end{cases}$$

4) adaptar o vetor de pesos $w(t+1) = w(t) + \eta [d(t) - y(t)] x(t)$.
$$d(t) = \begin{cases} +1 \text{ se o vertor de entrada} \in C1 \\ -1 \text{ se o vetor de entrada} \in C2 \end{cases}$$
onde $C1$ e $C2$ são classes.

5) Incremente o tempo por 1 e vá para o passo 2.

A rede na figura 8 é composta por um único neurônio, entretanto é possível construir um perceptron com vários neurônios na camada, sendo que o processo apresentado para aprendizagem de um neurônio, será expandido para todos os neurônios na camada.

Adaline/Madaline

O modelo ADALINE (ADAptive LINear Elements), Figura 9, e MADALINE (Multiple ADALINEs) foi construído por Bernard Widrow e Marcial Hoff. O modelo é um sistema adaptativo que aprende mais rapidamente e com maior precisão que o perceptron. A idéia consiste em introduzir limiares nos neurônios da rede. Assim, nestas redes um neurônio artificial faz a soma dos pesos de uma entrada multiplicada pelos pesos correspondentes, com um valor denominado bias (polarização), cuja função é antecipar ou retardar a ativação do neurônio, em função do seu sinal. Se a soma for maior que 0, a saída é +1, se a soma for menor ou igual a zero, a saída é -1. A seguir é apresentada a estrutura básica da rede.

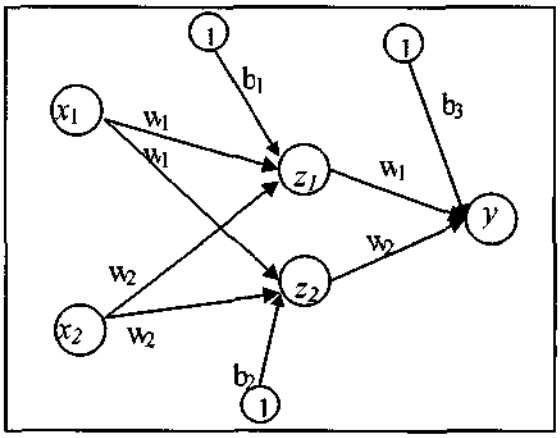

Figura 9 - Uma rede madaline com 2 neurônios adaline na camada escondida e um na camada de saída.

Backpropagation

Backpropagation é um método sistemático para o treinamento das redes neurais artificiais de múltiplas camadas. O algoritmo de treinamento foi apresentado por Rumelhart, Hinton e Willians em 1986, representando um passo para que as redes neurais fossem implementadas em situações práticas do mundo real. O algoritmo também foi desenvolvido por Werbos em 1974 e posteriormente por Parker em 1982. O algoritmo de backpropagation apresenta muitas vantagens em relação às redes de uma ou duas camadas. Estima-se que

aproximadamente 80% das aplicações com redes neurais utilizam o algoritmo de backpropagation.

Antes do desenvolvimento do algoritmo de backpropagation, as tentativas de usar perceptrons com mais de uma camada de pesos foram frustradas pelo que foi chamado de "weight assignment problem" (por exemplo, como alocar o erro da camada de saída entre duas camadas de pesos quando não há um fundamento matemático firme para se fazer isso). Esse problema limitante para aplicações de redes neurais.

Em geral a função de ativação utilizada em redes neurais treinadas por backpropagation é do tipo sigmoid ou tangente hiperbólica (Figura 4 e Figure 10).

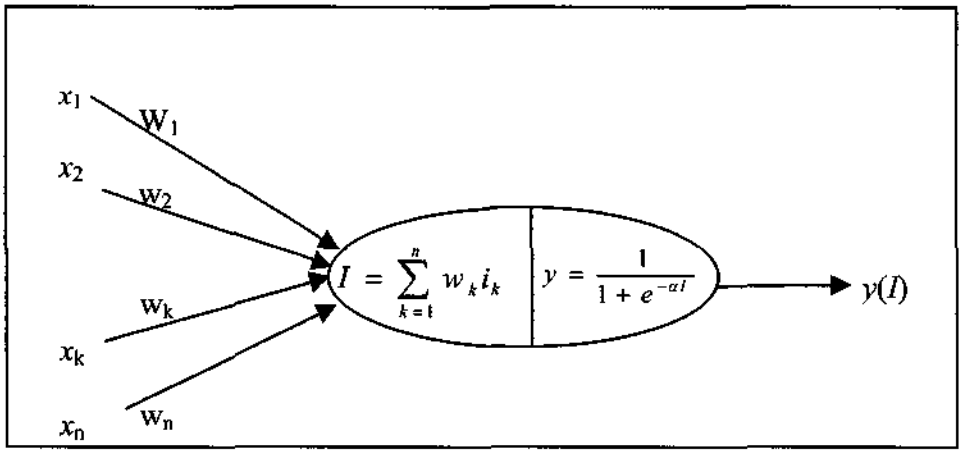

Figura 10 - Neurônio típico com função de ativação não linear.

A soma das entradas vezes o peso é dada pela equação 1, e a função de ativação usada sigmoid é dada por:

$$y(I) = \frac{1}{(1+e^{-\alpha I})} = (1+e^{-\alpha I})^{-1} \tag{2}$$

cuja a derivada é obtida por:

$$\frac{\partial y(I)}{\partial I} = (-1)(1+e^{-\alpha I})e^{-\alpha I}(-\alpha) \tag{3}$$

Isolando $e^{-\alpha I}$ da equação (2), substituindo na equação (3) e simplificando obtém-se:

$$\frac{\partial y(I)}{\partial I} = \alpha \frac{1-y(I)}{y(I)} y^2(I) = \alpha[1-y(I)]y(I) \tag{4}$$

Esta derivada da função é utilizada no cálculo da variação dos pesos que deve ser adicionada aos pesos da rede no processo de aprendizagem.

A introdução de não-linearidades nas redes de camadas múltiplas fazem com que estas tenham maior poder de representação do que as redes de uma única camada.

Mapas Auto-organizáveis

Tuevo Kohonen, um engenheiro elétrico da Universidade de Tecnologia de Helsinki, construiu uma rede auto-organizável que "aprende" sem que tenha sido determinada uma resposta correta para um padrão de entrada. Seu modelo é bastante próximo ao sistema neurobiológico. De forma análoga ao nosso sistema visual que traça um espaço visual na superfície do córtex visual, a rede traça representações do sinal analógico no seu conjunto de respostas de saída.

Essa rede auto-associativa tem uma camada única, recorrente e altamente conectada. Os pesos têm que ser inicializados e normalizados e as entradas têm que ser normalizadas ou ajustadas para alguma referência padrão. A regra de aprendizado é competitiva, o nó com resposta mais alta e sua vizinhança têm os pesos ajustados. Com o passar do tempo, o tamanho da vizinhança é reduzido. A vizinhança começa a ficar semelhante nas suas propriedades de resposta, e a organização global toma forma.

A rede Kohonen tem a habilidade de mapear a distribuição estatística, entretanto, funciona bem somente com redes grandes. Embora elas necessitem auto-treinamento, elas são muito rápidas e podem trabalhar em tempo real. Isso significa que a rede pode aprender continuamente e se adaptar às mudanças com o passar do tempo. Em aplicações estatísticas ou de modelamentos topológicos as entradas podem ser freqüências, localizações espaciais, e assim por diante. Mapas auto-organizantes pode ser mais eficientes do que muitos algoritmos na realização de cálculos.

2.6. Treinamento de uma Rede Neural de camadas múltiplas por Backpropagation

O objetivo do treinamento é ajustar os pesos para que a aplicação de um conjunto de entradas produza saídas desejadas. O processo de treinamento se dá da se seguinte forma:

1- os pesos são inicializados com pequenos valores aleatórios (positivos ou negativos) para assegurar que a rede não fique saturada por grandes valores de pesos (se todos os pesos fossem inicializados com valores iguais e o desempenho necessitasse de pesos diferentes, a rede poderia não treinar);

2- seleciona-se um par de treinamento do conjunto de treinamento;

3- o vetor de entrada é aplicado à entrada da rede;

4- calcula-se a saída da rede;

5- calcula-se o erro como a diferença entre a saída da rede e a saída desejada;

6- os pesos da rede são ajustados de forma a minimizar o erro, com a variação dos pesos dada por:

$$\Delta w_{ij}(n) = -\eta \frac{\partial E(n)}{\partial w_{ij}(n)} \tag{5}$$

$$\frac{\partial E(n)}{\partial w_{ij}(n)} = -e_j(n)\varphi'(v_j(n))y_i(n) \tag{6}$$

como $\delta_j = -e_j(n)\varphi'(v_j(n))$ é o gradiente local ao neurônio j.

Logo,

$$\Delta w_{ij}(n) = \eta \delta_j(n) y_i(n) \tag{7}$$

onde $E(n)$ = é o erro calculado na saída da rede, η é a taxa de aprendizagem, $v_j(n)$ é calculado como $v_j(n) = \sum_{i=1}^{n} w_{ji}(n) y_i(n)$, com $y_i(n)$ sendo uma entrada para o neurônio j, que é fornecida pela neurônio i na camada antecedente. $\varphi'(v_j(n))$ é calculada pela equação (4)

7- os passos 2 a 6 são repetidos para cada par dos vetores de entrada/saída

O treinamento da rede neural envolve dois passos: o passo "forward" (para frente) quando os sinais de entrada propagam-se da entrada para a saída da rede. No passo de retropropagação, os erros calculados são propagados de volta através da rede sendo usados para ajustar os pesos. A saída de uma camada é a entrada para a próxima. Os pesos da camada de saída são ajustados primeiro, usando a modificação pela regra delta. A regra delta também é conhecida como regra de Widrow ou método do gradiente, obtém o ponto mínimo através de um processo de iteração local, utilizando um exemplo do conjunto de treinamento por vez. Seja o erro quadrático $E(n)$:

$$E(n) = (y_d(n) - y(n))^2 \tag{5}$$

Onde $y(n)$ é o vetor obtido da rede e $y_d(n)$ o vetor com a saída desejada. O método do gradiente parte de um ponto arbitrário $w(0)$, e pode caminhar pela superfície E em direção ao ponto de mínimo, evoluindo sempre no sentido oposto ao do gradiente naquele ponto.

Em seguida são ajustados os pesos das camadas escondidas. O problema é que as camadas escondidas não têm valores alvos. Consequentemente, o treinamento é mais complicado, porque o erro tem que ser propagado de volta à rede, camada por camada.

Ao contrário do modelo de Hopfield, a backpropagation pode armazenar uma porcentagem muito maior de padrões. Ela generaliza bem e pode realizar arbitrariamente mapeamentos não lineares. Uma das limitações existentes é que a rede requer muitos treinamentos supervisionados, com muitos exemplos de pares de entrada/saída, e não há garantia de convergência.

Essencialmente a rede backpropagation emprega o refinamento da técnica de Widrow-Hoff, que calcula a diferença entre as saídas atuais e as saídas desejadas. Usando esses erros os pesos são modificados na proporção do erro do sinal de entrada (retro-propagação do erro).

As redes neurais têm sido aplicadas no desenvolvimento de sistemas voltados para tarefas de navegação robótica como em Sethi e Yu, (1994) e Jochem e Pormeleau, (1996).

3. LÓGICA NEBULOSA

A informação imperfeita advém de imprecisão, conflito, ignorância parcial, etc. sobre certos fatos e dados. Entretanto, mesmo considerando a imperfeição é necessário tratá-la para tomar uma decisão.

A teoria dos conjuntos nebulosos é um dos modelos que tratam a informação imprecisa. O modelo foi introduzido por Zadeh em 1965, com o objetivo de permitir graduações da pertinência de um elemento a uma dada classe, ou seja de possibilitar a um elemento de pertencer com maior ou menor intensidade àquela classe. Isso é feito quando o grau de pertinência de um elemento ao conjunto, que na teoria dos conjuntos assume apenas os valores 0 ou 1, passa a ser dado por um valor no intervalo dos números reais [0,1].

Para um dado conjunto de discurso X, um subconjunto nebuloso A de X é definido por uma função pertinência que associa cada elemento x de X o grau $\mu_A(x)$, cujo valor varia entre 0 e 1, representando o grau com qual x pertence a A.

$$\mu_A : X \rightarrow [0,1]$$

Como exemplo considere a Figura definição de indivíduo alto.

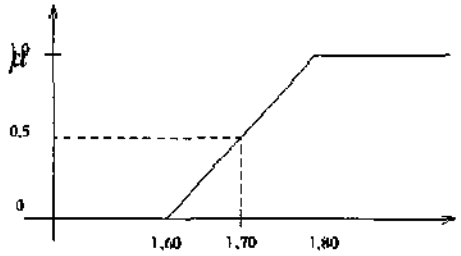

Figura 11 – Definição do conjunto nebuloso indivíduo alto.

A Figura 11 define um indivíduo alto como um conjunto nebuloso. No caso, quando um indivíduo tiver um 1,80 m ou mais ele é um indivíduo alto com certeza (μ =1). No caso de um indivíduo medir 1,60 m ou menos, definitivamente ele não é considerado um indivíduo alto (μ =0). Os indivíduos que têm altura entre 1,60 m e 1,80 m, serão considerados indivíduos altos com graus de pertinência diferentes (μ). Por exemplo, se um indivíduo com altura de 1,70 m pertence ao conjuntos dos altos com grau de inclusão de 0.5.

Desta forma os conjuntos nebulosos constituem uma "ponte" no caminho de aproximar o raciocínio humano ao da lógica executada pela máquina.

O grau de pertinência é naturalmente subjetivo sendo mais um problema de definição do que de medição e o número dado por μ, por exemplo, para o caso de um indivíduo alto deve ser encarado como a resposta da questão: "José tem 1,72 m. A que grau, na escala entre 0 e 1, o rótulo 'indivíduo alto' se aplica a ele?". O grau de pertinência não é probabilidade, mas uma medida da compatibilidade do objeto com o conceito representado pelo conjunto nebuloso. Logo, μ mede a compatibilidade do indivíduo José com a definição do conjunto nebuloso de indivíduos altos.

Assim como na teoria clássica, existem operadores para os conjuntos nebulosos. A negação é uma função que leva um elemento do conjunto [0,1], no conjunto [0,1], ou seja,

$$n : [0,1] \rightarrow [0,1]$$

podendo ser gerada por várias operações, sendo que a mais comum é:

$$n(x) = 1 - x \tag{6}$$

Com isso, para um conjunto nebuloso A, $\bar{A}$ representa a negação de A no universo X e o grau de pertinência de um elemento x do universo X é dado por .

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x) \tag{7}$$

A operação de interseção é implementada por uma família de operações denominadas *T-normas* e a união é implementada por uma família de operações denominadas *T-conormas*. O conjunto de *T-normas* e *T-conormas* é denominado conjunto de normas triangulares, sendo que cada norma triangular é uma função

$$v : [0,1] \times [0,1] \rightarrow [0,1]$$

que apresentam as seguintes propriedades:

1. Comutatividade: $v(x, y) = v(y, x)$
2. Associatividade: $v(x, v(y, z)) = v(v(x, y), z)$
3. Monotonicidade: se $x \leq z$ e $y \leq t$ então $v(x, y) \leq v(z, t)$
4. Elemento neutro para *T-normas*: $\top(x,1) = x$ (elemento neutro 1)

   Elemento neutro para *T-conormas*: $\perp(x,0) = x$ (elemento neutro 0)

A tabela 1 mostra algumas das *T-normas* e *T-conormas* mais utilizadas em sistemas que utilizam lógica nebulosa.

Tabela 1 – Exemplos de *T-normas* e *T-conormas* utilizadas na lógica nebulosa.

T-norma	*T-conorma*	*Nome*
$\min(x, y)$	$\max(x, y)$	Zadeh
$x.y$	$x + y - xy$	Probabilista
$\max(x + y - 1,0)$	$\min(x + y,1)$	Lukasiewicz
$\dfrac{xy}{\gamma + (1-\gamma)(x + y - xy)}$	$\dfrac{x + y - xy - (1-\gamma)xy}{1-(1-\gamma)xy}$	Hamacher ($\gamma > 0$)
x, *se* $y = 1$ y, *se* $x = 1$ 0, *se não*	x, *se* $y = 0$ y, *se* $x = 0$ 1, *se não*	Weber

Uma outra operação importante na lógica nebulosa é a *implicação* que é uma função como

$$I : [0,1] \times [0,1] \rightarrow [0,1]$$

Os operadores de implicação são usados para modelar as regras de inferência do tipo "Se (*premissa*) então (*conclusão*)". Se um valor α o grau de pertinência entre as condições estabelecidas na premissa e os valores encontrados na realidade e seja C, definido no universo Z, o conjunto nebuloso presente na conclusão da regra. Então, para verificar o grau de pertinência com a premissa implica a conclusão, dados os valores encontrados na realidade, verificaremos o quanto α implica $\mu_C(z)$, verificando $I(\alpha, \mu_C(z))$ para todo $z \in Z$.

A tabela 2 mostra exemplos de operadores de implicação mais comuns na literatura.

Tabela 2 – Exemplos de operações de *implicação* na lógica nebulosa.

Implicação	Nome
$\max(1-x, y)$	Kleene
$\min(1-x+y,1)$	Lukasiewicz
$1, \quad se \quad x \leq y$ $y \quad se \quad não$	Gödel
$\min(x, y)$	Mandani
$x.y$	Larsen

O uso mais significativo da lógica nebulosa em sistemas baseados em conhecimento, tem sido em sistemas de controle nebulosos (Tsoukalas e Uhrig, 1997). Um controlador nebuloso pode ser visto como um sistema especialista simplificado, onde a conseqüência de uma regra não é aplicada como antecedente de outra. Isto porque as ações de controle são baseadas em um único nível de inferência. As regras de controle nebuloso são usualmente do tipo:

$$\text{Regra } j\text{: Se } x_1 \text{ é } A_{1j} \text{ e } \ldots \text{ e } x_n \text{ é } A_{nj} \text{ então } y \text{ é } B_j$$

onde A_{ij} e B_j são conjuntos nebulosos, e a implicação é implementada como uma função de implicação nebulosa, como aquelas da tabela 2. Em cada ciclo do processo, os valores das variáveis x_i são medidos e então comparados aos conjuntos nebulosos A_{ij} nas regras, gerando uma medida da adequação dos valores medidos à premissa da regra (utilizando uma *T-norma* para implementar o conectivo "e" na premissa das regras, significando uma interseção das premissas). A implicação então utiliza esta medida de adequação e o conjunto nebuloso B_j na conclusão para obter um valor B'_j para a variável de controle, em relação àquela regra. Os valores B'_j são então agregados em uma única ação de controle C, utilizando uma *T-norma* ou *T-conorma*. Em alguns controladores os B'_j e C são nebulosos, então faz-se necessário se determinar um valor preciso para a variável de controle, a partir de C.

Um exemplo de sistema de navegação que utiliza lógica nebulosa pode ser encontrado em Lu e Wang, (1997).

4. EXEMPLOS DE SISTEMAS EXISTENTES

Já existem diversos trabalhos na literatura que implementam técnicas de navegação autônoma utilizando diversos tipos de sensores que fornecem informações do ambiente para

que a tarefa de controle seja executada. Neste capítulo são apresentados alguns trabalhos consultados no decorrer da pesquisa bibliográfico.

Jochem e Pomerleau (1996) descrevem a evolução de um projeto de pesquisa de aprendizagem de máquinas, para a criação de um sistema com direção robusta capaz de executar táticas de manobras, tais como a navegação da mudança e da interseção da pista ao longo da trajetória a ser percorrida.

No sistema, a navegação autônoma inclui o planeamento da rota, a passagem por obstáculos e a estimativa da posição, fazendo com que o robô mantenha-se na área delimitada pela visão da pista, utilizando uma câmera que extrai informações da estrada, a partir de linhas e bordas. O sistema de controle que mantém o robô na pista é baseado em visão, onde os operadores procuram características, tais como regiões e marcas coloridas do asfalto da pista para determinar os limites da estrada. O sistemas trabalha bem, quando as características programadas que foram detectadas, são claramente visíveis, mas apresentam dificuldades quando estas características estão obscurecidas ou ausentes. Outras dificuldades ocorrem quando muda-se a aparência da estrada, por exemplo, de uma rua da cidade para uma estrada, o que exige que o sistema seja reprogramado para extrair características da nova estrada, o que é uma tarefa que consume muito tempo.

Para sanar essas dificuldades os pesquisadores optaram por um sistema rápido, com um supercomputador com capacidade de processamento paralelo, que se adaptasse às novas características das estradas, e construíram uma pista que permitisse a exploração da capacidade de aprendizagem. Utilizando-se do processamento paralelo do computador foi feito um algoritmo de aprendizagem baseado em backpropagation (ver seção 2).

Buhmamm, (1995) desenvolveu um robô denominado Rhino, que foi projeto como um robô móvel para navegação dentro de um ambiente, e para a manipulação de tarefas. O objetivo científico geral do projeto Rhino, é o desenvolvimento e análise de sistemas autônomos e aprendizado complexo. O artigo descreve a maioria dos componentes do software de controle do Rhino e a filosofia básica da arquitetura Rhino.

O Rhino possui 24 sensores de proximidade (sonar), um sistema de câmera montada em uma unidade paint-tilt e 2 computadores 486 on-board. A informação do sonar é obtida em uma taxa de 1.3 Hz e as imagens das câmeras são processadas a uma taxa de 0.7 Hz. O Rhino comunica-se com computadores externos (duas Sparcstation Sun) através de um link via internet. As características chaves do software do controle Rhino são autonomia, aprendizado, operação em tempo real, deliberação e controle reativo, comunicação e controle distribuição, comunicação assíncrona e software tolerante à falha.

Shem, (1997) apresenta o projeto Yoda da Universidade da Califórnia para criação de agentes móveis que podem autonomamente aprender com base nas suas próprias ações, percepções e missões. O planejamento da trajetória usa mapas para determinar a seqüência de metas a ser executada.

A navegação autônoma tem sido bastante utilizada na construção de robôs para competição e exibição em eventos como a copa de futebol jogado por robôs, ou Robo cup. A tecnologia de robô vem progredindo significativamente nos últimos anos e sensores e atuadores eficientes, junto com sistemas de potência, já estão disponíveis para uma gama de aplicações diferentes.

5. IMPLEMENTAÇÃO

O projeto em desenvolvimento tem como objetivo construir um sistema de navegação adaptativo para controlar um veículo em uma estrada ou pista, sem a intervenção humana.

A característica de adaptação para o sistema será implementada utilizando-se a lógica nebulosa e/ou redes neurais, para corrigir a trajetória do veículo de acordo com as informações adquiridas do ambiente.

A idéia inicial é atuar sobre o sistema a partir de informações extraídas de imagens capturadas por uma câmera. O sistema deverá localizar pontos ou objetos interessantes na imagem e tomar uma decisão, replanejando a seqüência de comandos a ser executada, de forma a mantê-lo na trajetória inicial.

A figura 1 mostra o diagrama geral do sistema em desenvolvimento. O veículo no sistema poderá ser implementado através de algum simulador, ou poderá ser um sistema móvel, como por exemplo um carro controlado remotamente. Apenas um sensor de imagem será utilizado, que em se tratando do simulador deverá apresentar frames em seqüência escolhidos de um conjunto de situações hipotéticas.

O sistema de IA basear-se-á em redes neurais, lógica nebulosa, ou em ambos os paradigmas. Para poder controlar o veículo utilizando algum destes paradigmas, é necessário escolher que dados serão utilizados para a tomada de decisão. Para isso, o sistema deverá navegar em um ambiente conhecido e controlado, com marcas luminosas ao longo do caminho, que serão usadas para fornecer as informações necessárias para a correção da trajetória, e que serão extraídas através de técnicas de visão computacional.

Os problemas com essa implementação estão relacionados com a escolha dos operadores de visão computacional que devem ser utilizados para extração de características,

com os modelos de redes neurais que devem ser utilizados, e com a formulação das regras nos sistemas ou subsistemas baseados em lógica nebulosa. Todos eles envolvem uma pesquisa para saber as vantagens em utilizá-los no desenvolvimento do sistema.

No caso de um sistema simulado, a interconexão entre as várias partes do sistema será realizada por módulos de software. Em se tratando de um sistema real, o sistema será dotado com uma câmera de vídeo, que será conectada a um sistema de vídeo link, que permitirá a transmissão por VHF para um equipamento receptor (um vídeo por exemplo), cuja saída será conectada à entrada de uma placa do tipo frame grabber (para captura de imagens).

O software de controle deverá dispor de um módulo para a captura das imagens, um módulo para a extração das características na imagens e um módulo para atuação sobre o sistema.

A atuação sobre o sistema de navegação no caso de simulação, será feita por algum módulo de sistema que incorpora o modelo do veículo observado. Em se tratando de um sistema móvel real, a atuação se dará em cima do controle remoto do veículo, evitando que este seja aberto para adequá-lo às necessidades do projeto. Para isso, será construído um circuito lógico que controlará o controle remoto, a partir de informações enviadas pela porta paralela do sistema computacional envolvido. Assim, o sistema como proposto, mostra uma total independência de conexões fixas por fio, sendo totalmente teleguiado.

Os paradigmas de IA mencionados anteriormente, poderão ser utilizados não apenas no desenvolvimento do módulo de controle, mas também nos módulos para extração de informações a partir das imagens, o que poderá resultar em um sistema híbrido que utiliza paradigmas distintos de IA.

Uma característica importante em se usar um sistema de veículo real comercial no desenvolvimento de uma tarefa de navegação autônoma, é a possibilidade de adaptação da metodologia de desenvolvimento para qualquer sistema móvel, de maneira rápida permitindo que os ajustes finos sejam executados pelo sistema de controle adaptativo.

6. CONCLUSÕES E PERSPECTIVAS

Este projeto propõe o desenvolvimento de um sistema adaptativo para a navegação autônoma de um veículo (ou simulador de veículo) controlado por controle remoto.

O sistema deverá executar as tarefas de controle utilizando técnicas de IA como lógica nebulosa e redes neurais. O sistema tomará decisões a partir de imagens capturadas por uma

câmera on-board, cujo sinal será transmitido por VHF para um sistema de captura conectado a um framegrabber.

O projeto está previsto para ser executado em até 18 meses conforme o cronograma em anexo. De acordo com este, um levantamento bibliográfico deve ser feito cobrindo um período de 10 anos, buscando subsídios para suportar as atividades previstas a serem desenvolvidas. A revisão bibliográfica ainda está em andamento juntamente com os estudos preliminares de redes neurais e lógica nebulosa.

O cronograma do projeto, considerando o que já foi realizado até o presente pode ser visto abaixo.

Cronograma de atividades:
1. Etapa 1- Estudo de técnicas de navegação, redes neurais e lógica fuzzy - 8 meses
2. Etapa 2- Modelagem de um método de navegação com controle inteligente– 5 meses
3. Etapa 3- Implementação e testes de um método de um modelo – 7 meses
4. Etapa 4- Relatório – 2 meses

Cronograma

Etapa	Mês 1	Mês 2	Mês 3	Mês 4	Mês 5	Mês 6	Mês 7	Mês 8	Mês 9	Mês 10	Mês 11	Mês 12	Mês 13	Mês 14	Mês 15	Mês 16	Mês 17	Mês 18
1	X **E**	X **E**	X **E**	X **E**	X	X	X	X										
2							X **E**	X	X	X	X							
3											X	X	X	X	X	X	X	
4																	X	X

X - Proposto

E – Executado

Em contato com as publicações que descrevem sistemas de navegação, foi concebida a idéia de como implementar a transferência de imagens para o sistema e como implementar a interface que controlará o controle remoto do sistema.

Os próximos passos de implementação serão o desenvolvimento do sistema de controle da navegação com técnicas de IA e a simulação de situações de navegação pré estabelecidas (como um plano de navegação) sem realimentação de informações do ambiente, com objetivo de calibrar o sistema para o controle do veículo ou simulador escolhido.

BIBLIOGRAFIA

Bittencourt, G., **Inteligência Artificial Ferramentas e Teorias**. Editora da UFSC, Florianópolis, 1998.

Brown, C. – Review of Artificial Intelligence and Mobile Robotics: Case Studies of Successful Robot Systems. **AI Magazine**, 19(4):141–144, Winter 1998.

Buhmann J., Burgard W., Cremers A.B., Fox D., Hofmann T., Schneider F., Strikos J. e Thrun S. "The Mobile Robot Rhino". **AI Magazine**, 16(2):31–38, Summer 1995.

Dean, T.; Bonasso, R.P., 1992 {AAAI} robot exhibition and competition. AI Magazine, 14(1):35-48, Spring 1993.

Haykin, S., Neural Networks: A Comprehensive Foundation. Mcmillan, NJ, 1994, p. 696.

Horn, R., **Robot Vision**. New York, McGraw Hill, 1986.

Jochem, T. and Pormeleau, D., Life in the fast Lane – The Evolution of an Adaptative Vehicle Contral System. **AI Magazine, 17(2):11–50, Summer 1996.**

Kitano H., Kuniyoshi Y., Noda I., Asada M., Matsubara H. e Osawa E. "RoboCup: A challenge problem for AI". **AI Magazine**, 18(1):73-85, Spring 1997.

Kosaka, A.; Pan, J., Purdue experiments in model-based vision for hallway navigation, Proceedings of Workshop on Vision for Robots in IROS'95, pp.87-96, 1995.

Kovács Z. L., **Redes Neurais Artificiais - Fundamentos e Aplicações**, Ed. Acadêmica 2ºed., R.J., 1996.

Li, W., Lu, G. e Wang, Y. "Recognizing White Line Markings for Vision Guided Vehicle Navigation by Fuzzy Reasoning". **Pattern Recognition Letters** 18(8):771-780, August 1997.

Lisboa, P. G. J. **Neural Networks**: Current Applications. Liverpool: Chapman & Hall, 1992. 279 p.

Meng, M.; Kak, A. C., Mobile robot navigation using neural networks and nonmetrical environment models, **IEEE Control Systems**, pp.30-39, October 1993.

Nadler M. e Smith E. P., **Pattern Recognition Engineering**, John Wiley & Sons Inc., 1ª Ed. Nova York, 1992., 588 p.

Nair, D.; Aggarwal, J. K., Detecting unexpected obstacles that appear in the path of a navigating robot. In Proc. First IEEE International Conference on Image Processing, volume 2, pages 311--315, Austin, TX, November 1994.

Osuma R. G. e Luo R. C., "Lola - Probabilistic Navigation for Topological Maps". **AI Magazine**, 17 (1):55-62, Spring 1995.

Pan, J.; Pack, D.; Kosaka, A.; Kak, A. C., FUZZY-NAV: A vision-based robot navigation architecture using fuzzy inference for uncertainty-reasoning, Proceedings of the World Congress on Neural Networks, Vol.2, pp.602-607, Washington D.C., July 17-21, 1995.

Pao, Y. H. **Adaptive Pattern Recognition and Neural Networks**. Addison-Wesley Publishing Company Inc., 1989. 309 p.

Sargent R., Bailey B., Witty C. e Wright A. "Dynamic Object Capture Using Fast Vision Tracking". **AI Magazine** 18(1):65–72., Spring 1997.

Sethi J. K. e Yu, G., "Making a Mobile Robot Learn to Determine its Location Using Neural Networks". **Pattern Recognition Letters** 15(4):393-402, April 1994.

Shen W.M., Adibi J., Cho B., Kaminka G., Kim J., Salemi B. e Tejada S., "YODA The Young Observant Discovery Agent". **AI magazine** 18(1):37-45, Spring 1997.

Talluri, R.; Aggarwal, J. K., Autonomous navigation in cluttered outdoor environments using geometric visibility constraints. In Proc. of the Intl. Conf. on Intelligent Autonomous Systems, IAS-3, February 1993.

Thrun S., "To Know or Not to Know - On the Utility of Models in Mobile Robotics". **AI Magazine** 18(1): 47-54, Spring 1997.

Tsoukalas L.H. e Uhrig R.E., **Fuzzy and Neural Approaches in Egineering**. John Wiley & Sons, 1997, p. 587.

Tyengar S. S. e Thomas D., "Autonomous Mobile Robot Research at Lousiana State University's Robotics Research Laboratory". **AI Magazine**, 13 (2):25-32, Summer, 1992.

Yang H. S., Chung J., Ryu B. S. e Lee J., "Cair 2 - Intelligent Mobile Robot for Guidance and Delivery". **AI magazine**, 17 (1):47-53, Spring 1995.

ANEXO

Cronograma de atividades para continuação do projeto

1) Etapa 1- Estudo de técnicas de navegação, redes neurais e lógica fuzzy - 8 meses
2) Etapa 2- Modelagem de um método de navegação com controle inteligente– 5 meses
3) Etapa 3- Implementação e testes de um método de um modelo – 7 meses
4) Etapa 4- Relatório – 2 meses

Cronograma

Etapa	Mês 1	Mês 2	Mês 3	Mês 4	Mês 5	Mês 6	Mês 7	Mês 8	Mês 9	Mês 10	Mês 11	Mês 12	Mês 13	Mês 14	Mês 15	Mês 16	Mês 17	Mês 18
1	X	X	X	X	X	X	X	X										
2							X	X	X	X	X							
3											X	X	X	X	X	X	X	
4																	X	X